Audience

The intended users of this document are the users with administrator privileges to the Workflow system.

Dependencies

The *Workflow* system is mostly built in *Java 1.5.x* and *Perl 5.6.x* **,** hence they are required to use this system. It requires *log4j* and *log4Perl* libraries to support cross-language logging services, *Castor*, *Xml-rpc* nd *JDOM* libraries to support XML operations and the *JUnit* libraries for validating the installation.

The workflow distribution comes with all the required libraries except the perl related modules. Following is the list of perl modules that are required by the Workflow system. These modules can be found and installed from www.cpan.org.

- Log::Log4Perl
- Config::IniFiles
- Archive::Any::Tar

If the Workflow system is being installed with a Grid support, then it also depends on two other scripts epilog and prolog, which come as part of the distribution and exist under the bin directory. These scripts have to installed on all the queues of the SGE, which may execute jobs that are submitted
through the Workflow. The prolog and epilog entries in the SGE queue configuration shoud look like below:

```
prolog:     <Path>/prolog $host $job_owner $job_id $job_name $queue
epilog:     <Path>/epilog $host $job_owner $job_id $job_name $queue
```

Deployment

The Workflow system is distributed through '***wf-n.n-src..tar.gz***' file. This distribution includes all the required third party libraries except JRE and Perl libraries. Refer to the Dependencies section for the details on JRE and Perl dependencies.

To deploy the Workflow system from the source :

- Download the latest distribution 'wf-n.n.src.tar.gz' from Sourceforge
- Expand the distribution file by using command '*tar zxvf wf-n.n.src.tar.gz'. This should create a directory called 'workflow' with all the source files in it.*
- Change directory into 'workflow', where the distribution file has been expanded.
- Create a tar distribution for installation purpose by using command 'ant create-dist'. This will create 'wf.tar' file under 'dist' directory.
- Expand this 'wf.tar' file contents to a temporary directory. This directory should have a script called '*deploy.sh*'. Change permissions of this script to make it an executable, if necessary.
- Invoke the 'deploy.sh' by '*./deploy.sh*'. This script prompts the user for the various required variables and modifies the installation files accordingly. The default *deployment directory* would be the current working directory. Following is the list of variables and their descriptions that can set during deployment.
    - ♦ *DEPLOYMENT DIRECTORY*: This is the location where the Workflow is installed. The default deployment directory would be the current directory. The *WF_ROOT* environment variable must be set to to deployment directory. If the deployment directory is different from the default, then the entire distribution files will be moved to the deployment directory from

the current directory. the deployment directory gets created if doesn't exist during the deployment.
- ♦ *ID GENERATOR CLASS*: This java class is used by the Workflow to generate the ids of the Command and CommandSets. The default Workflow id generator class is: *org.tigr.workflow.common.FileIDGenerator,* which generates the workflow ids based on the last generated id value that is stored in the *workflow.config* file. User's could overwrite this if they have their own custom id generator Java API, may be a Database based Id generator or any others.
- ♦ *GRID TYPE*: This is the name of the grid software, if used any by the the user. Supported grid types are '*condor*' and '*sge*' (SunGrid). If no grid support is required, skip this option.
- ♦ *SUPPORT TYPE*: This variable is only prompted when a grid support is requested. The two kinds of support types for a grid are:
    - ◊ *local* : Jobs are submitted to the grid from the local machine. This is default value.
    - ◊ *server*: Jobs are submitted to the grid through htcservice, if installed or applicable.
- ♦ SGE ROOT : This is the location where SGE is installed. Default value is '/local/n1ge'.
- ♦ SGE CELL : This is the primary SGE CELL name. Default value is 'default'."
- ♦ *HTC_BASE*: This is the directory where the htcservice is installed, if applicable. The environment variable *HTC_BASE* must be set to this value. This variable is only prompted when a server suport is requested for grid submissions.
- ♦ *MOCK_SERVER_BASE*: This is the directory where the all the distributed job related scripts and event log files are kept. This directory should have world read/write permissions with sticky (unix permissions) bit turned on as the ownership of the files that are generated here could belong to any user.

- Source either *exec_env.tcsh* or *exec_env.bash* file to start using the Workflow tools. These scripts are responsible for setting the PATH and WF_ROOT environment variables appropriately, which are necessary for using the Workflow system.

Testing

The Workflow distribution comes with a tool called *'RunTestSuite'* validates the newly deployed workflow system. *RunTestSuite* runs all the necessary tests against the installed Workflow system.

Summary of Command Line Options for *RunTestSuite*

Param	Brief Description	Required	Default	Valid values
mode	Specifies the interface to run the tests	Required		swing, text, awt
log	Log level	Optional	*3, Info*	0,1,2,3,4,5,6
logfile	Log file to be used	Optional	*stdout*	
delete	A *true* value for this parameter indicates that data that is created during the tests needs to be deleted. A *false* value indicates that the data generated need not to be deleted.	Optional	*true*	true, false
truncate	A *true* value for this parameter indicates that data that the tables that were created or updated during the tests need to be truncated at the end of the tests. A *false* value indicates that the same database	Optional	*true*	true, false

	tables do not need to be truncated the end of the tests.			
help	Prints a brief help message	Optional		

Details

mode: This parameter sets the interface to run the tests. The valid modes are *swing*, *awt* and *text.* It is a required parameter.

log: This parameter is used to specify a log level. This is an *optional* parameter All log messages logged at levels at or below the specified
log level will be logged to the file. If the user does not specify the log level explicitly, a default log level of 3 is used. There are 6 different
log levels that can be specified, see the logging section of the Workflow User Guide for details.

delete: This parameter indicates whether to delete the data created during the tests or not, at the end of the tests. 'By default, all the data created during the tests is deleted. It is an optional parameter.

truncate: This parameter indicates whether to truncate the tables that are created or updated during the tests or not, at the end of the tests. By default, all the tables that created/updated during the tests are truncated. It is an optional parameter.

logfile: This parameter is used to specify the name of the log file. If this parameter is not specified, program sends log messages to *stdout*.
This is an *optional* parameter.

help : This option produces a short help / usage message before exiting the program.

Usage and Examples

Usage:

```
RunTestSuite mode=<swing|awt|text> [log=<LogLevel (default 3)>]
                [delete=<true|false (default true)>]
                        [truncate=<true|false (default true)>]
                                [logfile=<log file name>] [--help]
```

Examples:

```
RunTestSuite mode=swing log=4 delete=false truncate=false logfile=test.log
```

Configuration / Properties files

Certain aspects of the Workflow system are configured by a number of configuration and properties files that are part of Workflow distribution. Following table lists these files with brief descriptions and the location of files.

File	Brief Description	location
workflow.config	This is the main configuration file used by the Workflow system. It configures several different variables with right paths to the processor, dispatcher look up files and mapping files, sets the thread throttling limits and sets the database server information.	Deployment directory.
log4j.properties	The logger configuration file used by the Workflow system.	Deployment directory.
log4perl.properties	The logger configuration file used by the perl components of the Workflow system.	Deployment directory.
command_proc_factory_lookup.prop	A lookup up file used to locate the appropriate CommandProcessor name.	*properties* directory under the deployment directory.
dispatcher_factory_lookup.prop	A lookup file used to locate the appropriate CommandDispatcher name.	*properties* directory under the deployment directory.
distributable_command_lookup.prop	A lookup file used to check if a Command is a distributable command or not.	*properties* directory under the deployment directory.
log4j_server.conf	The log4j configuration file used by the htcservice.	*server-conf* directory under the deployment directory.
sge_mockserver.conf	Configuration file to support job submissions 'SGE' grid.	*server-conf* directory under the deployment directory.

Thread Regulation

Each Command or CommandSet of a workflow in a Workflow engine is executed either by a CommandProcessor or a CommandDispatcher. Each CommandProcessor or CommandDispatcher is in turn run by a corresponding thread. When executing large workflows, creating and executing large number of concurrent threads may pose a risk of crashing the workflow engine or the host systems due to certain software / hardware limitations. To avoid this risk, the Workflow engine employs a mechanism to enforce a maximum limit on the number of concurrent Command Processors or Dispatchers that can run with in a workflow engine. It also restricts the number of concurrent DCE jobs that can be submitted to the grid by a Workflow engine.

These limits can only be changed by the workflow administrator by editing the file '*workflow.config'*. The appropriate values for these limits may be determined by the administrator based on the type and the size of workflows that are executed using the Workflow engine.

Following are the various limits that are enforced in the current release:

- **Command Processor Limits**
    - ♦ **max.proc.pool.count:** This limit represents the maximum number of concurrent CommandProcessors that can run inside a Workflow engine. For example: a value of 100 for this limit implies that at any given instance, only a maximum of 100 Commands can be executed with in the scope of a workflow engine.
    - ♦ **min.proc.pool.count:** This limit represents the number of threads that will be initialized in the processor pool during the Workflow initialization. For example: a value of of 10 for this limit implies that the Workflow engine initializes the processor thread pool with 10 threads at the startup. If required, more threads will be created in the increments of 1.
- **Command Dispatcher Limits**
    - ♦ **max.cmd.set.disp.count:** This limit represents the maximum number of concurrent CommandDispatchers that can run with in the scope of a CommandSet. For example: a value of 5 implies that if a CommandSet contains more than 5 CommandSets, then only a maximum of 5 CommandSets can be executed at any given instance.
    - ♦ **min.disp.pool.count:** This limit represents the number of threads to be initialized in the dispatcher pool during the Workflow initialization. For example: a value of of 5 for this limit implies that the Workflow engine initializes the dispatcher thread pool with 5 threads at the startup. If required, more threads will be created in the increments of 1.
    - ♦ **max.disp.pool.count:** This limit represents the maximum number of CommandDispatchers that can run inside a Workflow engine. This limit is set to 1000000, to allow as many dispatchers as possible to run simultaneously with in a workflow engine. Do not change this to a lesser value than the current value.
- **DCE Jobs Limit**
    - ♦ **max.htc.jobs.count:** This limit represents the the maximum number of concurrent DCE jobs that can be submitted to the grid. Current limit is 100. This implies that at given instance there can only be a maximum of 100 jobs submitted to the grid by a workflow engine.

Reporting Problems

Please address all problems, and concerns about *workflow system* by sending mail to ***antware@tigr.org.***